HONEYWELL

DPS 6 TOOLKIT SERIES-I REFERENCE MANUAL

SOFTWARE

DPS 6

TOOLKIT SERIES-I REFERENCE MANUAL

SUBJECT

Description of the TOOLKIT Software Package

SOFTWARE SUPPORTED

This manual supports Release 1.0 of TOOLKIT under Release 3.0 and subsequent releases of the MOD 400 Executive.

ORDER NUMBER

HH06-00

June 1986

Honeywell

About This Manual

This manual is written primarily for system administrators, programmers, and systems analysts who are familiar with GCOS 6 MOD 400 use and terminology. TOOLKIT provides utilities, active functions, and COBOL subroutines to help increase system effectiveness and COBOL program efficiency.

This manual is divided into five sections and one appendix.

Section 1 gives an introduction to TOOLKIT, a description of its components, and an overview of the manual's organization. The section closes with a table identifying and briefly describing each TOOLKIT tool, with a reference to the section that contains the complete tool description.

Section 2 describes how to use TOOLKIT. New users of TOOLKIT are shown how to use the menus and help files to become familiar with the tools. More experienced TOOLKIT users are given a brief description of using TOOLKIT tools at the system level.

Section 3 contains alphabetically-organized descriptions of each TOOLKIT utility tool.

Section 4 describes each of the TOOLKIT active function tools, organized alphabetically.

Section 5 includes descriptions of all TOOLKIT COBOL subroutine tools.

Appendix A contains all COBOL copy files used by TOOLKIT's COBOL subroutines.

A Technical Publictions Remarks Form, included at the back of this manual, can be used to note any corrections, changes, or additions that you think would make the manual more useful.

File No.: 1R73 HH06-00

Contents

Figures

Tables

Section 1
Introduction to TOOLKIT Series-I

TOOLKIT Series-I is a software package designed to help you get the most of GCOS 6 MOD 400 resources. TOOLKIT provides:

- Group, system, record lock, buffer and I/O pool, and memory usage information, for tailoring the design of the operating system
- File, bound unit, and disk utilization information, for streamlining COBOL program design
- Filename and character string locating tools, for maintaining programs and searching for data
- Forms viewing, testing, and attributes listing tools, for working with single or multiple forms
- Namelist and buffer description creation tools, for simplifying the writing of programs that use VDAM
- Argument-passing subroutines, for including in COBOL programs so they can function as commands
- GCOS 6 MOD 400 System Services interface subroutines, for extending the power of COBOL programs
- Other handy tools for general use

TOOLKIT's interactive utilities are designed to work quickly and provide accurate, reliable results. These information-gathering commands produce logically-organized output in a condensed format for easy screen viewing. In addition, TOOLKIT provides Execution Control (EC) file tools (utilities and active functions) and COBOL subroutines designed for specific programming or user tasks.

TOOLKIT Components

Altogether, TOOLKIT provides fourteen utilities, two active functions, and four COBOL callable subroutines. TOOLKIT's menus and help files simplify learning each tool and its options. When you are familiar with TOOLKIT, you can use any tool directly at MOD 400 command level.

How This Manual Is Organized

This introductory section provides an overview of TOOLKIT and briefly describes all tools (see Tables 1-1 and 1-2) so you can see which ones may be of use to you. Note that the tables group tools according to their primary users (programmers and administrators) – this is not to imply that TOOLKIT places any restrictions on the use of its tools. Any user can use any tool for any task. You have to know the task, however, and you should already be familiar with MOD 400 conventions and operations.

Section 2 describes how to use TOOLKIT, both through the menu system (for learning to use tools) and at MOD 400 command level (for practical daily use once you are familiar with TOOLKIT).

The rest of the manual is divided into three sections, Utilities, Active Functions, and Callable Subroutines. (Tables 1-1 and 1-2 identify the section to which each tool belongs.) Within each section, tools are organized alphabetically.

Table 1-1. TOOLKIT Programmer's Tools

Function	Description	Tool	Section
Accessing System Services	Makes MOD 400 System Services accessible to COBOL programs without requiring programmer knowledge of Assembler	CMCL	Callable Subroutines
Clearing temporary files	Deletes the information in a file or series of files while leaving the file(s) intact	CLR	Utilities
Displaying bound unit attributes	Displays attributes for a user-specified series of bound units, including memory distribution analysis	LBUS	Utilities
Displaying file information	Displays file information (sector statistics, file fragmentation and hashing data); includes integrated WALK-SUBTREE and command driver	LIST	Utilities
Displaying mail waiting message	Returns the number of pending mail messages without actually displaying the mail	POST	Active Functions
Echoing mixed text and active functions	Allows the mixing of text and active functions on EC lines, to be echoed back later, with all active functions resolved	TYPE	Utilities
Locating character strings	Locates and displays all occurences found to match a user-specified character string, identifying directory name, filename, and line number of the string	SRCH	Utilities
Locating files	Returns the full path of all files found to match the user-specified filename or star name	HUNT	Utilities
Managing and using forms	Allows forms viewing, testing, and attributes listing for single or multiple forms (star names accepted); also allows creation of name lists and buffer descriptions for COBOL programs that use VDAM	FUM	Utilities
Passing arguments	Passes floating-option arguments to executing COBOL programs so they can function as commands	ARGTAB	Callable Subroutines
Passing arguments	Passes positional arguments to executing COBOL programs so they can function as commands	GETARG	Callable Subroutines
Removing LFNs	Removes, by force, all Logical File Numbers (LFNs) within the specified range	REMALL	Utilities
Reporting device usage	Displays snapshots of use and percentage of available space of all disk and tape devices, by volume	VOLT	Utilities
Returning volume ID	Returns the name of the volume mounted on the specified device	VOLLEY	Active Functions
Supplying pointers	Identifies the physical address of a field in memory	POINT	Callable Subroutines

Table 1-2. TOOLKIT Administrator's Tools

Function	Description	Tool	Section
Displaying mail waiting message	Returns the number of pending mail messages without actually displaying the mail	POST	Active Functions
Echoing mixed text and active functions	Allows the mixing of text and active functions on EC lines, to be echoed back later, with all active functions resolved	TYPE	Utilities
Monitoring overall system activities	Displays current group and system activities by overall system, series of groups, or a single group	SPY	Utilities
Monitoring pool usage	Displays detailed buffer and I/O pool statistics of current pool usage, for "tuning" pool size to meet needs	BIO	Utilities
Monitoring resource usage	Displays detailed resource use statistics, by group	RUM	Utilities
Reporting current activities	Displays a quick-look snapshot of who is currently using the system, and what bound units are being used	WHOM	Utilities
Reporting record lock usage	Displays record lock usage statistics, to be used for adjusting system record lock allocation, or for identifying programs that use too many locks	KEYS	Utilities
Returning volume ID	Returns the name of the volume mounted on the specified device	VOLLEY	Active Functions

Section 2
Using TOOLKIT Series-I

TOOLKIT Series-I is easy to learn and use. To get familiar with TOOLKIT's tools and their options, you can let the menu system guide you. After a while, it will be faster to use TOOLKIT directly at the command level, just as you would MOD 400 commands. This section explains both methods of use.

Note: For installation instructions, see the *TOOLKIT Software Release Bulletin* (Order No. HH01-00). TOOLKIT may also be initialized as a lead task; see *GCOS 6 MOD 400 System Building and Administration* (Order No. CZ02).

Using the TOOLKIT Menu System

To invoke TOOLKIT at the MOD 400 command level, enter:

```
TLKIT
```

Upon initialization, TOOLKIT will display the TOOLKIT Interactive Utilities Menu, shown in Figure 2-1.

```
Honeywell                                                    Ones &
Information                    ---------------               Zeros
 Systems                       ---------------          Copyright @1986
                                    TOOLKIT
                               _______________

                          INTERACTIVE UTILITIES

          Administration                          Programming
          SPY                                     FUM
          BIO                                     LIST
          RUM                                     LBUS
          WHOM                                    SRCH
          KEY                                     HUNT
                                                  VOLT

                          Enter TOOL Name:

        f1=Other Tools Info   f2=Command Line Input   f3=Exit
```

Figure 2-1. TOOLKIT Interactive Utilities Menu

The TOOLKIT Interactive Utilities menu lists the names of all tools that can be entered directly at the ENTER TOOL NAME prompt; three function key entries are listed at the bottom of the menu.

Entering Selections at the TOOLKIT Interactive Utilities Menu

To select an interactive tool, simply enter the first two characters of the tool name at the ENTER TOOL NAME prompt, then press the RETURN key. TOOLKIT will respond with a help message display for the tool selected.

For example, if you enter SPY at the Interactive Tools ENTER TOOL NAME prompt, TOOLKIT displays the help file and command line arguments entry prompt shown in Figure 2-2.

```
    SPY Selected...

-------------------------------------------------------------------------
SPY displays current group and system activities.
Command Syntax:               SPY [option option .... option]

Command Options: -L [nn]   Loop display every nn secs. Default is 6 secs.
                 -G [group_id group_id...group_id]  Display groups in list
                 -X [group_id group_id...group_id]  Exclude groups in list
                 -F         Display under FORMS mode (i.e. TCLP)
                 -C         Current cumulative values for I/O and CPU%
                 -D         Replace MEM and POOL with OFFSET and TRANS NAME
                 -R         Reset Disk I/Os and %CPU Totals
                 -T nn      Display nn times    -A   Display without paging
                  ?         Display help info   -Q   End SPY execution
-------------------------------------------------------------------------


Enter desired command line arguments  --

         SPY _____________________________________________________


                      f3=Return to TOOLKIT Menu
```

Figure 2-2. SPY Help File and Prompt Line

When the help file for the selected tool is displayed, the cursor is positioned next to the tool name in the command line, ready for you to enter any desired arguments (listed under "Command Options" on the help file screen).

For tools that have no required arguments, you may simply press the RETURN key to get the default display. If you do want to enter arguments on the command line, begin entering at the current cursor position and put a space between each argument. After the last argument, press the RETURN key. TOOLKIT will execute the tool's function.

Using the Function Keys at the TOOLKIT Interactive Utilities Menu

As previously mentioned, the bottom of the TOOLKIT Interactive Utilities menu lists three function keys: Other Tools Info, Command Line Input, and Exit. Each of these three functions is described below.

To select one of the functions, simply press the corresponding function key at your keyboard. TOOLKIT will execute the selected function.

To see the Other Available Tools menu, press the F1 key. TOOLKIT displays a menu of non-interactive TOOLKIT tools, as shown in Figure 2-3.

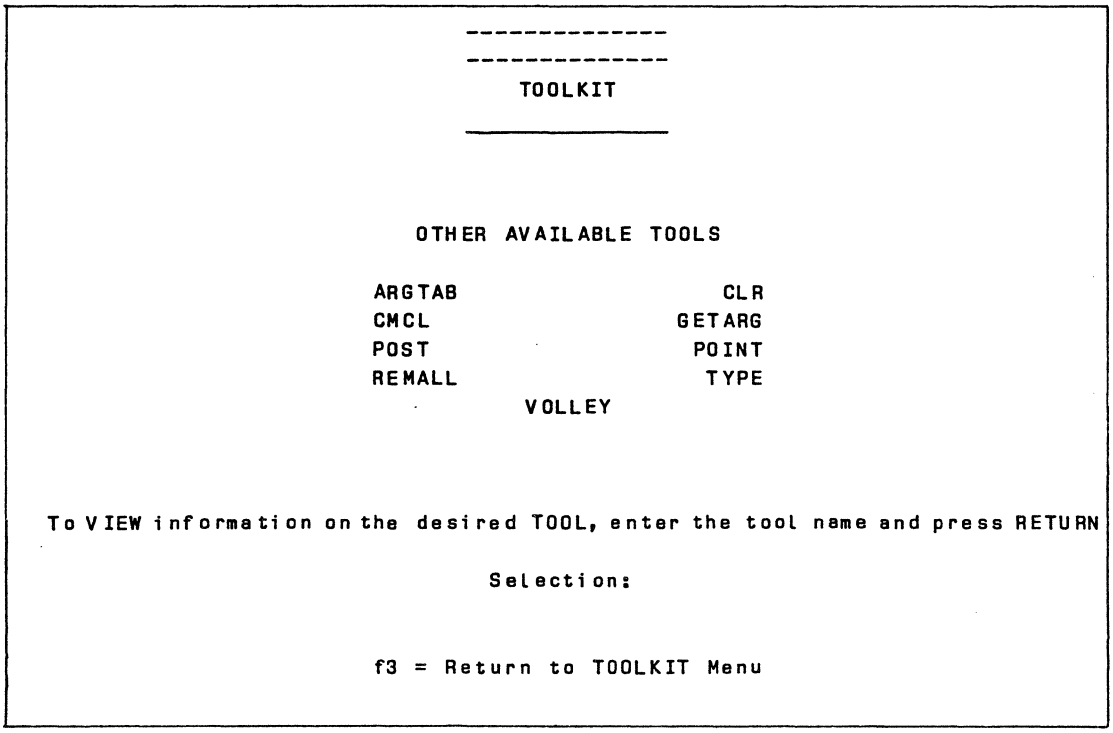


Figure 2-3. TOOLKIT Other Available Tools Menu

To get a display of help information about one of these non-interactive tools, enter the tool name at the SELECTION prompt, then press the RETURN key. TOOLKIT will respond with a brief description of the selected tool. When you have read the help information, press the RETURN key to return to the TOOLKIT Interactive Utilities menu. (Complete descriptions of these Other Available Tools are included in this manual.)

Because the tools listed on this Other Available Tools menu are non-interactive, you have to exit TOOLKIT in order to use them.

To enter MOD 400 (or TOOLKIT) commands without exiting from TOOLKIT, press the F2 key. The Execution Control Language (ECL) prompt will be displayed below the TOOLKIT Interactive Utilities menu:

```
Ecl :_____________________________________________
```

Enter the complete command line, then press the RETURN key. When command execution is complete, press the RETURN key to return to the TOOLKIT Interactive Utilities menu.

Note: If your ECL command line entry contained an error, command execution will fail. In this case, when you press the RETURN key to return to the TOOLKIT Interactive Utilities menu, the cursor returns to the ECL prompt and displays the command line you entered so you can try again. You can correct the command line at the ECL prompt, or press the F3 key to move the cursor to the ENTER TOOL NAME prompt.

If your ECL command line entry was correct, command execution will be successful. In this case, pressing the RETURN key causes the cursor to return to the ENTER TOOL NAME prompt at the TOOLKIT Interactive Utilities menu.

To exit the TOOLKIT menu system, press the F3 key.

Using TOOLKIT at the MOD 400 Command Level

Once you are familiar with TOOLKIT's tools and their options, you will probably prefer to use the tools directly at MOD 400 command level, without going through the TOOLKIT menu system. This is quite simple to do — enter the tool name and any arguments exactly as you would within the TOOLKIT menu system.

If you are unfamiliar with the tool's options and do not want to use the help file, refer to the tool descriptions given in the following sections of this manual. Each tool description shows the command syntax for invoking the tool at MOD 400 command level.

Section 3
Utilities

BIO

Function

Displays detailed buffer and input/output (I/O) pool statistics of current pool usage. BIO can be used to measure: the amount of disk I/O on all devices to determine device balancing, buffer pool utilization and performance, and the amount of disk I/O for each group in the system.

BIO requires some preparatory setup (see DEVICE STATISTICS and BUFFER POOL STATISTICS under "Notes").

For tips on how to analyze BIO statistics, see TIPS ON ANALYZING BIO OUTPUT under "Notes."

Command Syntax BIO [-L]

Command Options

?

Display HELP messages on command syntax.

-L

Update display every 6 seconds.

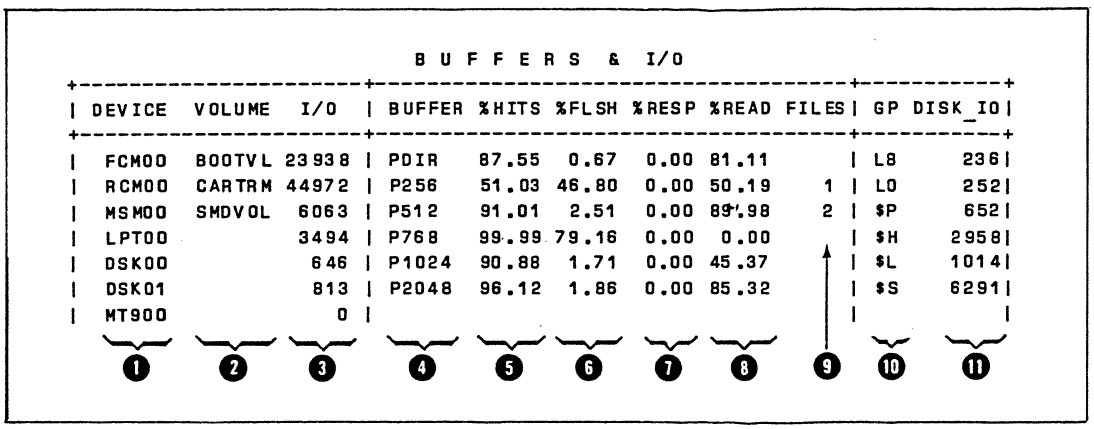
-Q

End execution of BIO (entered during execution).

Command Defaults Display once only

Display Example

Each numbered item in this example corresponds to a description in "Display Description."



```
                                B U F F E R S   &   I/O
+----------------------+-------------------------------------+-----------+
| DEVICE  VOLUME   I/O | BUFFER %HITS %FLSH %RESP %READ FILES| GP DISK_IO|
+----------------------+-------------------------------------+-----------+
|  FCM00  BOOTVL 23938 | PDIR   87.55  0.67  0.00 81.11      | L8      236|
|  RCM00  CARTRM 44972 | P256   51.03 46.80  0.00 50.19    1 | LO      252|
|  MSM00  SMDVOL  6063 | P512   91.01  2.51  0.00 89.98    2 | $P      652|
|  LPT00          3494 | P768   99.99 79.16  0.00  0.00      | $H     2958|
|  DSK00           646 | P1024  90.88  1.71  0.00 45.37      | $L     1014|
|  DSK01           813 | P2048  96.12  1.86  0.00 85.32      | $S     6291|
|  MT900             0 |                                     |          |
    ❶       ❷      ❸      ❹     ❺     ❻     ❼     ❽     ❾    ❿     ⓫
```

Display Description

Device Information

(1) Devices monitored by error logging

(2) Names of volumes mounted on the device

(3) Total disk I/Os since the last reboot

Buffer Pool Information

(4) General public buffer pool name

(5) Percentage of a disc block found in memory (so a physical I/O is not necessary)

(6) Percentage of a modified buffer that must be prematurely written to disk in order to free a buffer for the current request

(7) Percentage of times that a buffer reservation had to wait for a buffer to become available

(8) Percentage of times that a disc block is read versus the percentage of times it is written

(9) Number of open files that are using the specified buffer

I/O Information

(10) Group ID

(11) Total number of disk I/Os since the group began or since BIO was initiated (whichever was later)

Notes:

Device Statistics

To monitor device statistics, set up error logging:

1. Add to the CLM_USER file:

```
LDBU ZERRST
```

2. Add to the START_UP.EC file:

```
> >SYSLIB2>START_ELOG MEMORY
> >SYSLIB2>START_ELOG device_id (eg., MSM00)
```

for each device to monitor.
examples:

```
START_ELOG MEMORY
START_ELOG MSM00
START_ELOG RCM00
START_ELOG MT900
```

See the *MOD 400 System User's Guide* (Order No. CZ04) for more information on error logging.

Buffer Pool Statistics

To monitor buffer pools, change the START_UP.EC file. Add to the START_UP.EC file:

```
> >SYSLIB2>CBP poolname arguments
```

examples:

```
CBP P256 -BFSZ 256 -NBF 15
CBP P512 -BFSZ 512 -NBF 15
```

See the *MOD 400 Commands* manual (Order No. CZ17) for more information on buffer pools.

Tips on Analyzing BIO Output

1. %Hits should approach 100% in a perfect environment.
2. %Flush should stay below 10%.
3. %Response should stay below 1%.
4. If %Hits is below 50% and %Flush is below 10% you can probably decrease the number of buffers.
5. If %Flush is greater than 10%, add buffers.
6. If %Response is greater than 1%, add buffers.
7. Buffer pool statistics are measured from the last time the system was rebooted or ">>SYSLIB2>BPI -RESET" was executed. To measure the effect of a particular structured test it is recommended to reset the statistics. Also, to get an idea of total buffer pool utility, measure the system several times over a day's activity.

Resource Requirements

MEMORY	Shared procedure	1004 words
	Data	120 words
CPU	Varies (DPS model 75 uses 2.1%)	
CIP	Required (simulator use acceptable)	
DISK I/O	None	

CLR

Function

Deletes the information in a file or series of files while leaving the file(s) intact. CLR will empty from one to five files, retaining all file attributes. A typical use is to clear temporary files used by an execution control (EC) process.

Command Syntax CLR filename1 filename2 ... filename5

Command Options

filename

Any MOD400 file pathname is valid.

Command Defaults None.

Resource Requirements

MEMORY:	Shared procedure	74 words
	Data	32 words

FUM (Forms Utility Manager)

Function

Allows forms viewing, testing, and attributes listing for single or multiple forms; also allows creation of namelists and buffer descriptions for COBOL programs that use VDAM. Accepts star names.

Command Syntax FUM form_name [process_option] [control_options]

Command Options

?

Display HELP messages on command syntax.

form_name

Any form name or star name.

process_option

[-L \-P \-T \-V \-E \-C] [form_file_path]

-L

List form names and form attributes (revision number, date/time updated, number fields, size, screen, block mode, field mode, buffer).

-P

Print form(s) as they would appear on the CRT.

-T

Test form(s) for valid attributes.

-V

View form(s) on CRT.

-E

Erase form(s) from forms_file.

-C

Create copy files needed by COBOL programs.

form_file_path

Use alternate forms file other than > >FORMS>FORMS.

control_option

-I

Information option. Used with print option (P) to display column numbers at top and bottom.

Used with copy option (–C) to display the current form that is being processed as FUM goes through the forms_file.

Used with the erase option (–E) to request verification from the user before erasing each file.

Used with the print option (–P) to print a column/row border around the form.

-O out_path

Change the user_out pathname. Eliminates the need to execute the file_out (FO) command.

-D copy_dir

Change the output directory for copy files from >>LDD>INCLUDE.

-G [R\W\N]

Create specific copy files.

R

Create NAMELIST and BUFFER copy files for fields which can be both input and output (variable entry fields).

W

Create NAMELIST and BUFFER copy files for fields which can only be updated by the application (constant fields).

N

Create NAMELIST and BUFFER copy files for non-display fields.
The default is to create NAMELIST and BUFFER files for all of the above.

Syntax:

```
-G
-G R
-G W
-G N
```

The filenames created for –G and its various options are:

Option	Filename	File Type
No Option (Default)	Fform.IN.C	NAMELIST
	Bform.IN.C	BUFFER
R	Rform.IN.C	NAMELIST
	Iform.IN.C	BUFFER
W	Wform.IN.C	NAMELIST
	Oform.IN.C	BUFFER
N	Nform.IN.C	NAMELIST
	Xform.IN.C	BUFFER

where 'form' is the last six (6) characters of the form_name.

Command Defaults

form_name	All forms found in forms_file
forms_file	>>FORMS>FORMS
copy_dir	>>LDD>INCLUDE
process_option	–L

An entry of 'FUM' with no parameters will produce a complete list of all forms in >>FORMS>FORMS.

Command Line Examples

FUM	List all forms in >>FORMS>FORMS
FUM A*Z	List all forms that start with "A" and end with "Z"
FUM ???	List all forms with 3 character names
FUM * –L >MYFORM	List all forms in the forms_file "MYFORM"
FUM A* –P –O !LPT00	Print forms that start with "A" on the printer
FUM ANYFORM –T	Test form "ANYFORM"
FUM * –T	Test all forms
FUM * –C	Create copy files for all forms

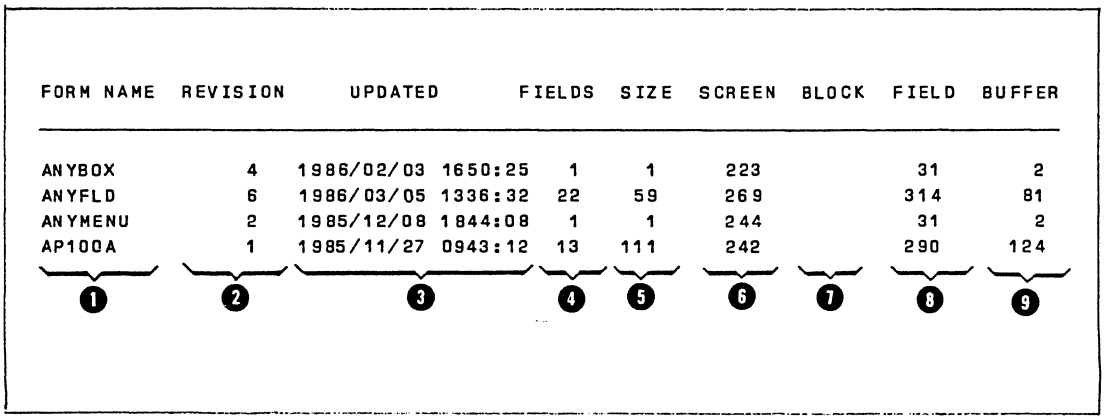
Generated COBOL Copy File Examples

```
01 BB-RELATION.
   03 B-NAME          PIC X(30) VALUE SPACE.
   03 B-ADDRESS       PIC X(30) VALUE SPACE.
   03 B-ADDRESS       PIC X(30) VALUE SPACE.
   03 B-CSZ           PIC X(29) VALUE SPACE.
   03 B-PHONE         PIC X(10) VALUE SPACE.
```

```
01 FN-RELATION.
03 F-NAME          PIC X(05) VALUE "NAME ".
03 F-ADDRESS       PIC X(05) VALUE "ADD1 ".
03 F-ADDRESS       PIC X(05) VALUE "ADD2 ".
03 F-CSZ           PIC X(04) VALUE "CSZ ".
03 F-PHONE         PIC X(06) VALUE "PHONE ".
03 FILLER          PIC X(01) VALUE ".".
```

Display Example

Each numbered item in this example corresponds to an entry in "Display Description."



FORM NAME	REVISION	UPDATED	FIELDS	SIZE	SCREEN	BLOCK	FIELD	BUFFER
ANYBOX	4	1986/02/03 1650:25	1	1	223		31	2
ANYFLD	6	1986/03/05 1336:32	22	59	269		314	81
ANYMENU	2	1985/12/08 1844:08	1	1	244		31	2
AP100A	1	1985/11/27 0943:12	13	111	242		290	124
❶	❷	❸	❹	❺	❻	❼	❽	❾

Display Description

(1) Name of the form in the forms file

(2) Number of times the form has been updated

(3) Date/time of last update

(4) Number of variable fields on form

(5) Field size total (in words)

(6) Size of screen descriptor

(7) Size of block-mode screen descriptor (in words)

(8) Size of field descriptors (in words)

(9) Size of terminal buffer

Resource Requirements

MEMORY	Unshared procedure/data	11268 words

HUNT

Function

Returns the full pathname of all files found (on a volume) to match the specified filename or star name.

Command Syntax

HUNT filename

Command Options

filename

Any MOD 400 filename or star name.

Command Defaults

None.

Example

An entry of:

```
HUNT *.EC
```

Might produce:

```
^BOOTVL>USER1>START_UP.EC
^BOOTVL>USERA>START_UP.EC
^BOOTVL>USERA>COMPILE.EC
^BOOTVL>START_UP.EC
^BOOTVL>SET_DATE.EC
...etc.
```

Resource Requirements

MEMORY	Unshared procedure/data	904 words

KEYS

Function

Displays record lock usage statistics, to be used for adjusting system record lock allocation, or for identifying programs that use too many locks.

Command Syntax KEYS [option ... option]

Command Options

–L [nn_seconds]

Loop every nn seconds (–L with no argument defaults to looping every 6 seconds)

–T nn_intervals

Loop nn times

–A

Auto-no paging

–Q

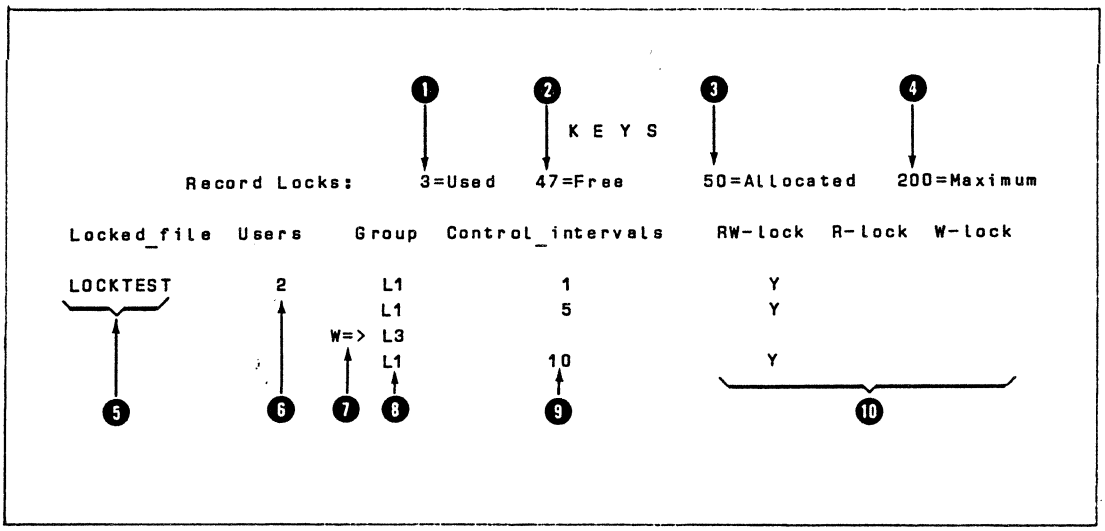
Terminate execution

Command Defaults Display once

Paging

Display Example

Each numbered item in this example corresponds to an entry in "Display Description."



Display Description

(1) Number of RECORD LOCKS used while KEYS was in execution.

(2) Number of RECORD LOCKS free.

(3) Number of RECORD LOCKS allocated.

(4) Maximum number of RECORD LOCKS.

(5) Name of file with outstanding lock.

(6) Number of users of locked file.

(7) Identification of group waiting for the lock (indicated by W=>).

(8) Group ID.

(9) Control interval numbers locked.

(10) Lock type: 'Y' (Yes) or blank (No) for the following:
Read-Write Lock
Read Lock
Write Lock

Resource Requirements

MEMORY	Shared procedure	732 words
	Data	80 words
CPU	Varies (DPS 6 model 75 uses 1.5%)	
CIP	Required (Simulator use acceptable)	
DISK I/O	None	

LBUS

Function

Displays attributes for the specified series of bound units, includes a memory distribution analysis. LBUS will search a path (directory > filename) for bound units matching the filename or star name.

Command Syntax LBUS [pathname] [option option]

Command Options

–A

Auto (printer mode)

?

Display HELP messages on command syntax

The following options are mutually exclusive; LBUS will accept the last option entered:

–C

Sort by creation date

–D

Sort by data size

–P

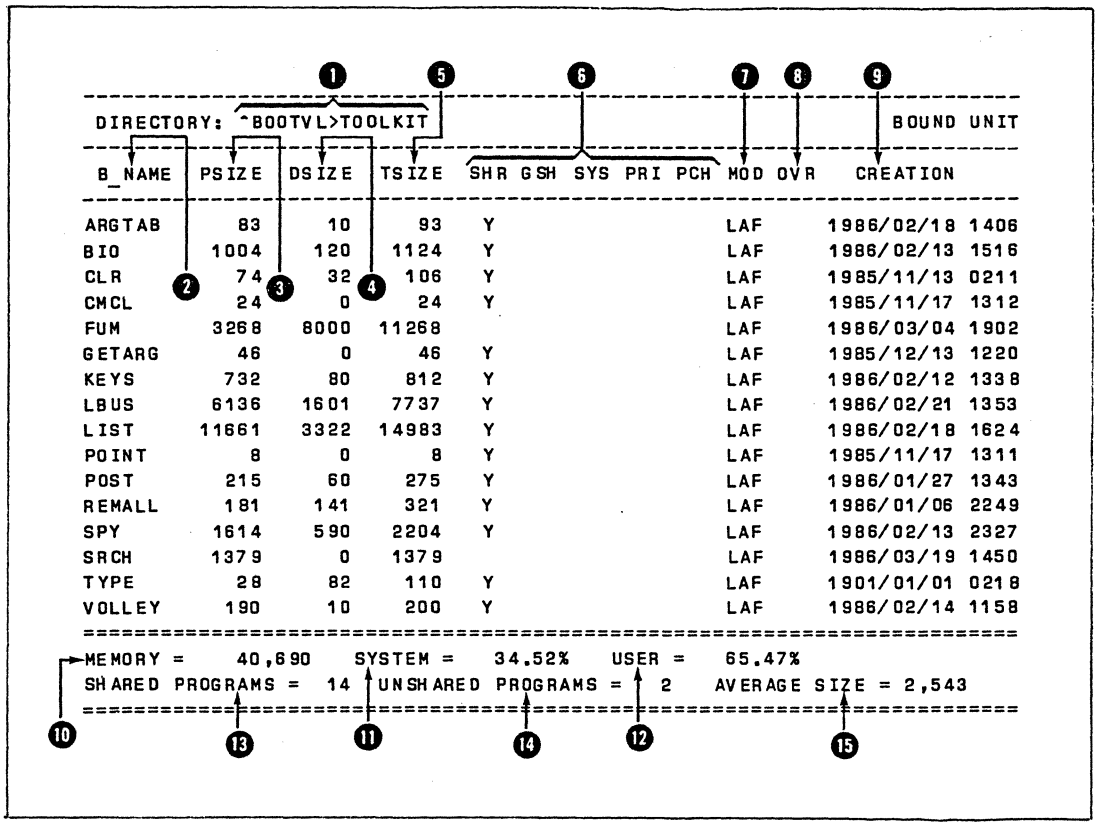
Sort by procedure size

–T

Sort by total size

Display Example

Each numbered item in this example corresponds to an entry in "Display Description."



Display Description

1. Directory name
2. Internal bound unit name
3. Procedure size
4. Data size
5. Total program size (procedure + data)
6. 'Y' (yes) or blank (no) for the following:
   - Share bit
   - Global share bit
   - System bit
   - Privilege mode
   - Patched
7. Long address/short address format (LAF/SAF)
8. Number of overlays
9. Creation date/time
10. Total memory (total of TSIZE column)

(11) Percentage of SYSTEM POOL usage

(12) Percentage of USER POOL usage

(13) Number of shared programs

(14) Number of unshared programs

(15) Average program size in words

Resource Requirements

MEMORY:	Shared procedure	6136 words
	Data	1601 words
CIP	Required (simulator use acceptable)	

LIST

Function

Displays file information (sector statistics, file fragmentation and hashing data); includes integrated WALK-SUBTREE and command driver. Accepts star names.

Command Syntax LIST [pathname] [option option ... option]

Command Options

pathname

[directory >]filename or star name. Pathnames are expressed as:

Directory > filename ... filename

LIST supports a maximum of 10 filenames.

?

Display HELP messages on command syntax

-T

List specific file organization types. The valid types are:

ALT, DIR, DYN, F_R, IND, INX, IDS, REL, RAN, SEQ, FILES

If the -T option is not used, all types are displayed. This option can also be used in combination with the other options to provide additional selection criteria.

-BF

Brief display. Displays name of file, type, number of sectors*.

-FULL

Full display. Displays name of file, type, logical record size (LRSZ), control interval size (CISZ), number of sectors, number of sectors used, number of extents*.

-CISZ

Control interval size. Displays file name, control interval size (CISZ)*.

-LRSZ

Logical record size. Displays file name, logical record size (LRSZ)*.

-N

Name only. Displays file names only*.

-XT

File extensions. Displays file name, number of extents*.

*Options BF, FULL, CISZ, LRSZ, N, and XT are mutually exclusive.

-S

Sort the files by display data. Major key is selected option (or file type if no option is selected); minor key is file name.

-SR

Sort major key in descending order.

-ECL 'Ecl string'

used to execute a MOD400 command for each file LIST encounters. Use % for the substitution of filename and %% for the substitution of full pathname.

For example: LIST ALL.* -ECL 'PF %'

This will find any file in the current working directory with a filename of ALL and any extension, and perform a Peruse File on each of these.

-ACK

Used in conjunction with the ECL option to request an acknowledgment of each file BEFORE performing the command string in the ECL option. LIST will display the filename and prompt you for execution of the command. Valid responses are:

Y – execute command
Q – stop processing
Use any other key to skip execution of command string.

-WS

Walk the directory structure and display against each directory occurrence. This option will gather up all directories subordinate to the initial directory, sort them in ascending sequence and then begin the list process. This option will also produce grand totals for all the listed directories (see Totals Display below).

-GTO

Display grand totals only. No detail information is displayed.

-P

Print mode. Sets the line length to 132; expands the number of entries on a display line. When creating a print file (FO filename), the Auto Mode option should be used in conjunction with this option.

If the ECL option is used with the Print option, the command is written out to the user-out file instead of being executed.

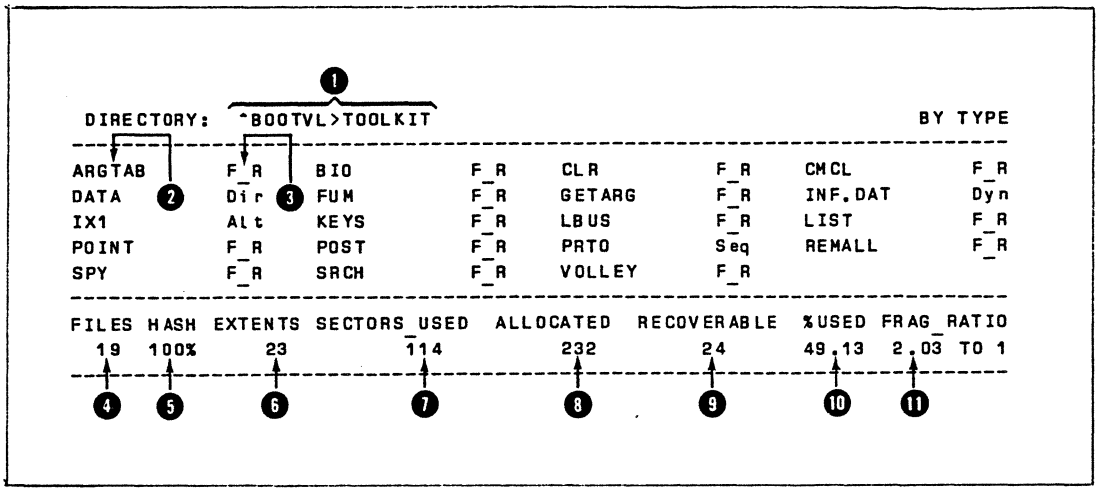
-A

Auto mode. Prevents the "More ?" question appearing after each page.

Command Defaults Displays current directory's file information
Displays all file organization types
No grand totals

Display Example

Each numbered item in this example corresponds to a description in "Display Description."



Display Description

(1) Directory Name

(2) Filename (of each file in the directory)

(3) File Type (of each file in the directory)

Totals Display

Totals are displayed after each invocation of LIST for each directory searched; grand totals are displayed when the walk-subtree option (WS) is selected. The total line consists of the following:

(4) Total number of files

(5) Hashing (efficiency of file distribution)

(6) Total number of file extensions

(7) Total number of sectors used within allocated number

(8) Total number of allocated sectors

(9) Total number of sectors which would be given back to the available sector pool if a SHRINK command were executed.

(10) Percentage of used / allocated sectors

(11) Ratio of extensions to files (fragmentation ratio)

Notes

The responses to the MORE? question are

Q Quit
T Quit and terminate walk subtree
A Switch to AUTO mode
Any other character to continue

Resource Requirements

MEMORY:	Shared procedure	11661 words
	Data	3322 words
CIP:	Required (simulator use acceptable)	

REMALL

Function

Removes, by force, all Logical File Numbers (LFNs) within the user-specified range. Removing by force affects both open and closed files.

Command Syntax REMALL [first_lfn [last_lfn]]

Command Options

First_lfn and last_lfn are optional:

If a first_lfn only is entered, REMALL will remove only one LFN.

Command Defaults If no options are entered, removes all reserved LFNs within the user's current group.

Command Line Examples and Descriptions

REMALL 23

Remove reserved LFN from file 23, even if that file is open

REMALL 23 45

Remove all reserved LFNs from files 23 through 45, from both open and closed files

Resource Requirements

MEMORY:	Shared procedure	181 words
	Data	140 words

RUM

Function

Displays detailed resource use statistics, by group (to identify heavy pool use).

Command Syntax RUM [option option .... option]

Command Options

?

Display HELP messages on command syntax

–L [nn]

Loop display every nn seconds. Default is 6 seconds.

–G [group id group id...group id]

Display only those groups in list.

–X [group id group id...group id]

Exclude groups from display.

–T nn

Display nn times.

–A

Display without screen paging.

–F

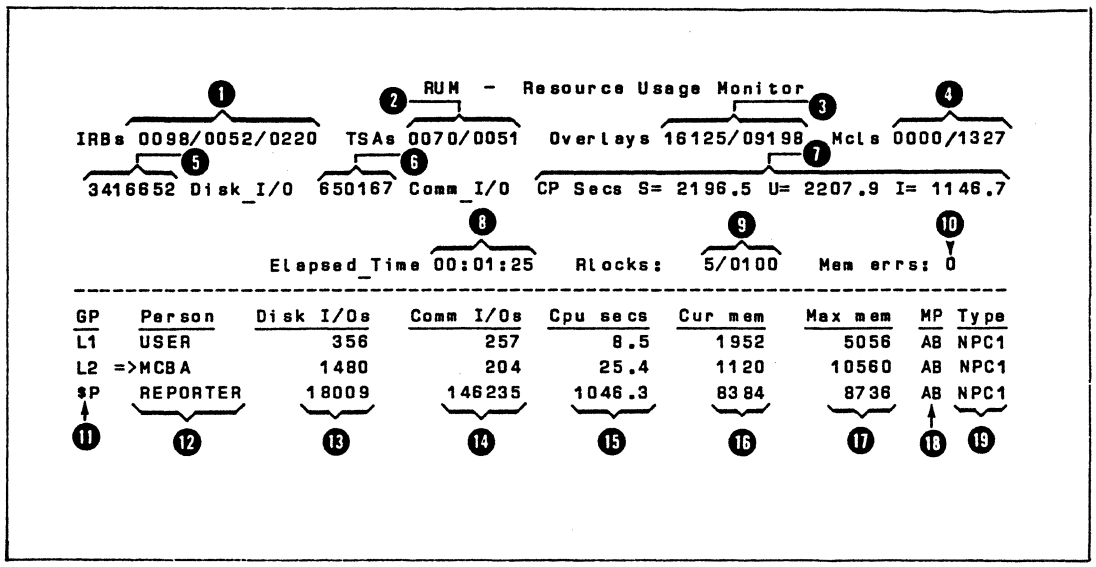
Display under FORMS mode (i.e., Transaction Control Language Processing: TCLP).

–Q

End execution (entered during execution).

Display Example

Each numbered item in this example corresponds to a description in "Display Description."



Display Description

Overall Volume Statistics

(1) Intermediate Request Blocks (IRBs): Maximum/Minimum/Allocated

(2) Trap Save Areas (TSAs): Current/Minimum

(3) Overlays: Requested/Loaded

(4) System Macro calls (MCLS):
Without stack expansion/With stack expansion

(5) Total number of disk_I/Os since RUM was invoked

(6) Total number of comm_I/Os since RUM was invoked

(7) CP Seconds: System/User/Idle

(8) Time elapsed since RUM was invoked (HH:MM:SS)

(9) Maximum record locks used while RUM executing/number allocated

(10) Number of memory errors logged

Group Statistics

(11) Group ID

(12) User name

(13) Total physical disk I/Os since RUM was invoked

(14) Number of comm I/Os since RUM was invoked

(15) Total CPU seconds since RUM was invoked

(16) Amount of memory used by bound unit (decimal)

(17) Peak amount of memory used by group

(18) Memory pool ID

(19) Memory pool type; these are:

X – exclusive
S – swappool
B – batch
N – non-exclusive
P – protected
C – contained
I – independent
The number under the 'type' heading is the execution ring of the pool.

SPY

Function

Displays current group and system activities by overall system, a series of groups, or a single group. Typical uses of SPY include:

- Measure the effects and resources of a particular program.
- Snapshot of the current system users.
- Measure the progress of a particular user.
- Provide Transaction Control Language Processing (TCLP) users with a tool to show system usage.

Command Syntax SPY [option option .... option]

Command Options

?

Display HELP messages on command syntax.

–L [nn]

Loop display every nn seconds. Default is 6 seconds.

–G [group_id group_id...group_id]

Display only those groups in list.

–X [group_id group_id...group_id]

Exclude groups from display.

–T nn

Display nn times.

–A

Display without screen paging.

–F

Display under FORMS mode (i.e., TCLP).

–C

Current cumulative values for I/O and CPU%.

–D

Debug mode, to replace memory and pool usage data with data for program offset and transaction name for TCLP.

–Q

End execution (entered during execution).

–R

Reset Disk I/Os and %CPU Totals (entered during execution).

Command Defaults Displays data for all groups on the system

Data averaged over life of execution

Pages screens

Displays memory and pool usage data (and does not display program offset and transaction name)

Display Example

Each numbered item in this example corresponds to an entry in "Display Description."

```
   1          2           3       4            5          6    7        8     9
   ↓          ↓           ↓       ↓            ↓          ↓    ↓        ↓     ↓
 GROUP |   PERSON      |LVL|  DIRECTORY  |DISK IOS| PROGRAM/MEM   | %CPU  |POOL
 ------------------------------------------------------------------------------------
   L7    |ADRIA        | 32| DEV         |        | PRO     34189 |       |AB NPC1  36363
   L8 => |KEN          | 32| KENK        |       9| SPY      2204 |  6.81 |$$ XCO   26624
   LO    |OPE          | 34| OPERATOR    |        | OPER          |       |
 M $P    |ADRIA        | 34| DEV         |        | DMON          |       |
   $H    |HIS          | 32| POTTER      |        | AMT           |       |
   $L    |LISTEN       | 41|             |        | LISTEN        |       |
   $S    |OPERATOR     | 29| POTTER      |        | Z3EXEC  26624 |       |
```

Display description:

(1) State and Group ID. States are:

space	active
M	mount pending
X	aborted (waiting on open I/O)
S	suspended
D	dormant

(2) User name

(3) Task level

(4) Current working directory

(5) Total physical disk I/Os since SPY was invoked or reset

(6) Bound unit currently in use

(7) Amount of memory used by bound unit (decimal).

(8) Percentage of cumulative CPU time since SPY was invoked or reset

(9) Pool usage – Pool ID
Pool type
Memory usage

Note: in 'debug mode' (option D), Memory and Pool usage are replaced by the following:

Program – relative program offset.

Transaction – transaction name if TCLP is being used.

Resource Requirements

MEMORY:	Shared procedure	1614 words
	Data	590 words
CPU	Varies (2.5% on DPS6 model 75)	
CIP	Required (simulator use acceptable)	
Disk I/O	None	

Notes

1. System performance enhancements may be measured by recording statistics before and after changes.
2. SPY's output may be written to disk and used for logging (FO file name).
3. The disk I/Os are cumulative and zero-based; that is, when SPY begins it records the current number of disk I/Os the group has performed and subtracts this base number before each subsequent display. This is the number of physical I/Os which occur. If buffer pooling is in effect, the I/Os satisfied by memory blocks are not recorded.
4. CPU percentage is the percent of CPU time used over the available time since SPY began. This measurement will not show the system peaks but is used to measure the CPU usage over time. If this number begins to increase, the group is becoming more CPU intensive; if the number decreases, the group is becoming more I/O bound or is idle.

VOLT

Function

Displays snapshots of use and percentage of available space on all disk and tape devices, by (mounted) volume. Typical uses:

- to quickly determine where space is available for file growth
- to find out which volumes are mounted on system devices

Command Syntax VOLT

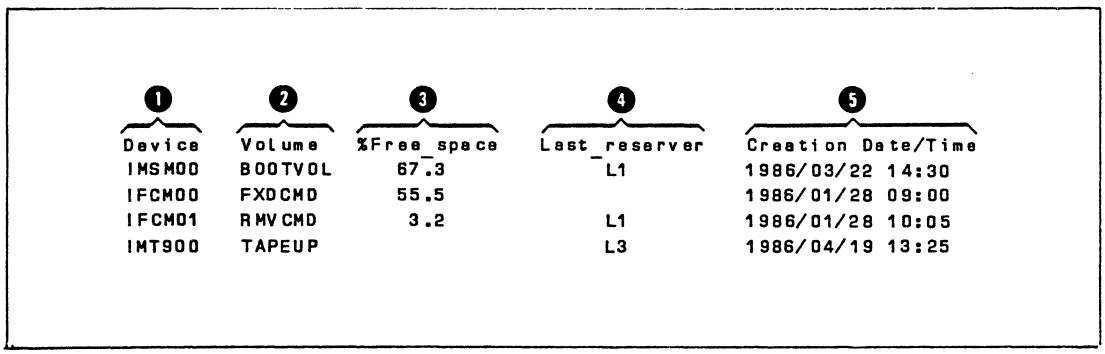
Command Options

None

Command Defaults N/A

Display Example

Each numbered item in this example corresponds to an entry in "Display Description."



❶ Device	❷ Volume	❸ %Free_space	❹ Last_reserver	❺ Creation Date/Time
IMSM00	BOOTVOL	67.3	L1	1986/03/22 14:30
IFCM00	FXDCMD	55.5		1986/01/28 09:00
IFCM01	RMVCMD	3.2	L1	1986/01/28 10:05
IMT900	TAPEUP		L3	1986/04/19 13:25

Display Description

(1) Device ID

(2) Name of mounted volume

(3) Percentage of free space

(4) ID of the last group to reserve the device

(5) Date and time the volume was created

Resource Requirements

MEMORY	Shared procedure	190 words
	Data	10 words

SRCH

Function

Given a character string or star name string, locates and displays all records that include the string. Display includes the directory name, filename, record line number, and actual record data. Typical uses for SRCH include:

- Find all programs in a COBOL library referring to a specific subroutine or FD (data file description).
- Find all Transaction Control Language Processing (TCLP) programs initializing a specific form.

Command Syntax SRCH [pathname] 'string' [option .... option]

Command Options

pathname

Directory > filename/star name.

string

Any valid ASCII string.

?

Display HELP messages on command syntax.

–G

Global search. Report all occurrences of string found.

–S

Silent mode. Report only on files containing string.

–F

Form feed. Issue page eject before each new file reported.

Command Defaults Lists all files searched.

Reports only the first occurrence found in each file.

Example

For an entry of:

```
SRCH COBOL_PRG 'CALL' -S -G
```

WHOM

Function

Quick-look snapshot of who is currently using the system, and what bound units are being used.

Command Syntax WHOM

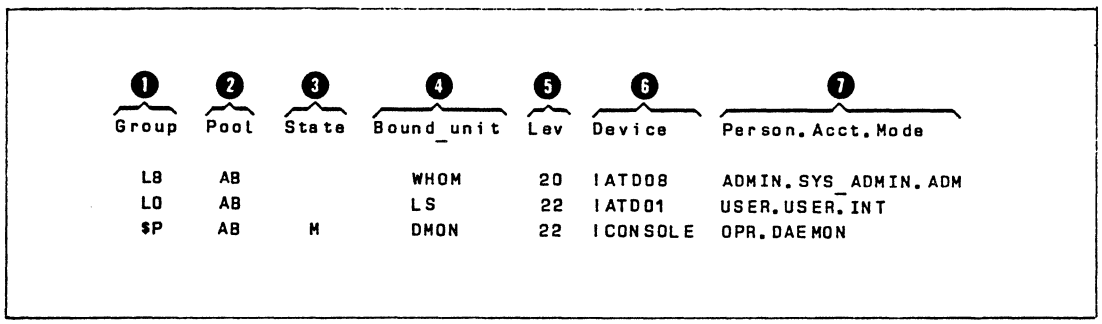
Command Options

None

Command Defaults N/A

Display Example

Each numbered item in this example corresponds to an entry in "Display Description."



❶ Group	❷ Pool	❸ State	❹ Bound_unit	❺ Lev	❻ Device	❼ Person.Acct.Mode
L8	AB		WHOM	20	!ATD08	ADMIN.SYS_ADMIN.ADM
L0	AB		LS	22	!ATD01	USER.USER.INT
$P	AB	M	DMON	22	!CONSOLE	OPR.DAEMON

Display Description

(1) Group ID

(2) Pool ID

(3) State of group; states are:

space	active
M	mount pending
X	aborted (waiting on open I/O)
S	suspended
D	dormant

(4) Name of bound unit currently executing

(5) Execution level

(6) User-in device ID

(7) User identification (Person.Account.Mode)

Resource Requirements

MEMORY:	Shared procedure	950 words

Section 4
Active Functions

POST

Function

Returns the number of pending mail messages for the current user, without actually displaying the mail. A typical use is to notify the user at login that mail messages have been received.

Command Syntax [POST]

Command Options

None

Command Defaults N/A

Example

An entry of:

```
>>SYSLIB2>TYPE 'You have ' [>>SYSLIB2>POST] ' mail messages'
```

could return:

```
You have 0 mail messages
```

Resource Requirements:

MEMORY	Shared procedure	215 words
	Data	60 words

VOLLEY

Function

Given a device name, returns the name of its mounted volume.

Command Syntax VOLLEY [device name]

Command Options

device name

Any valid device name.

Command Defaults N/A

Example

An entry of:

```
>>SYSLIB2>TYPE 'Boot volume is: ' [>>SYSLIB2>VOLLEY MSM00]
```

could return:

```
Boot volume is: MYVOL
```

Display Description:

(1) The VOLUME_ID if a volume is mounted.

(2) The string "Empty" if the device is valid but no volume is currently mounted.

(3) The string "Null" if the device doesn't exist.

Note: The returns "Empty" and "Null" are UPPER and lower case, as shown.

Resource Requirements:

MEMORY	Shared procedure	190 words
	Data	10 words

Section 5
Callable Subroutines

ARGTAB

Function

Passes floating-option arguments to executing COBOL programs so they can function as commands. Any number of arguments may be passed using a single call. (See GETARG for passing positional arguments.)

Subroutine Syntax

Add to the WORKING-STORAGE SECTION of your program:

```
01 OPTION-TABLE-SIZE.
   10  MAX-OPTIONS            COMP-1 VALUE n.

01 OPTION-TABLE.
   05  TABLE-ENTRY OCCURS n TIMES.
      10  THE-OPTION.
          20 THE-DASH         PICTURE X(001).
          20 SELECTION        PICTURE X(011).
      10  ARGUMENT-SIZE       COMP-1.
      10 THE-ARGUMENT         PICTURE X(060).
```

(Where "n" = the maximum number of options (programmer's choice))

Add to the PROCEDURE DIVISION of your program:

```
CALL "ARGTAB" USING OPTION-TABLE OPTION-TABLE-SIZE.
```

Note that ARGTAB may be called once only per execution of the COBOL program. Also, you can name the fields as you like, as long as you use the same name in the CALL statement.

Example

ARGTAB assumes that you have preceded each option with a dash (–). Options with a preceding dash are placed in the WORKING-STORAGE SECTION field called THE-OPTION. Arguments (which do NOT have preceding dashes) are placed in THE-ARGUMENT. The size of the argument is placed in ARGUMENT-SIZE. This is repeated for each argument in the command line.

For an entry of:

```
CBLPRG -L -T 10 -Z
```

(Assume that CBLPRG calls the ARGTAB subroutine.)

Using the WORKING-STORAGE section description shown above (in "Subroutine Syntax") ARGTAB passes the arguments into a table as follows:

SELECTION (1) will contain 'L'
SELECTION (2) will contain 'T'
THE-ARGUMENT (2) will contain '10'
ARGUMENT-SIZE (2) will contain '2'
SELECTION (3) will contain 'Z'

ARGTAB will also function properly if the program is a lead task.

CMCL

Function

Makes MOD 400 System Services directly accessible to COBOL programs. CMCL can use TOOLKIT's COBOL copy files (shown in the appendixes in the back of this manual) and can work in conjunction with the POINT subroutine.

Subroutine Description

The CMCL subroutine allows COBOL programs to execute any System Service Call described in *GCOS 6 MOD 400 Programmer's Guide – Volume II* (Order No. CZ06). This provides COBOL programs with the interface to MOD 400 services while eliminating the need to write customized assembly language subroutines to satisfy special requirements. It also provides more efficient ways to perform services than executing a command line and puts you in control of the outcome of a service request for better error handling.

Requirements for programs using MOD 400 System Services are explained below and illustrated in the example that follows.

Add to the WORKING-STORAGE SECTION of Your Program:

1. **Register Descriptions:** all programs that call system services must contain descriptions of DPS 6 "B" and "R" registers. TOOLKIT provides a complete description of these registers in the COBOL copy file REGS.IN.C (see Appendix A). If you include the statement:

   ```
   COPY REGS.
   ```

   in the WORKING-STORAGE SECTION of your program, REGS.IN.C will be copied from directory > >LDD>INCLUDE into your program during compilation.

2. **Data Descriptions:** each service call needs supporting data descriptions. The *GCOS 6 MOD 400 Programmer's Guide – Volume II* contains the data descriptions required for each system service call. TOOLKIT provides COBOL copy files for five commonly-used data descriptions (Create File Information, File Information Block, File Information, Get File Information, and Expand Path Table – see Appendix B). If you include a COPY statement (similar to the one shown in Point 1, above) in the WORKING-STORAGE SECTION of your program, the specified data description will be copied from directory > >LDD>INCLUDE into your program during compilation.

   If the service call requires data descriptions other than the five provided by TOOLKIT, you will have to enter the code (exactly as shown in the *GCOS 6 MOD 400 Programmer's Guide – Volume II)* in the WORKING-STORAGE SECTION of your program.

3. **Memory address pointers:** many service calls require pointers. TOOLKIT's POINT subroutine can provide the value of an address of any 01 level item in your program. To use the POINT subroutine, you must set up a WORKING-STORAGE SECTION data description for the memory address pointer.

4. **MCL description:** all service calls require an MCL description in the program's WORKING-STORAGE SECTION. The MCL description identifies the service call name and describes its data field. Appendix C includes the MCL descriptions of all MOD 400 System Service Calls.

Subroutine Syntax

Once the supporting register descriptions, data descriptions, and pointers (if needed) are included in the WORKING-STORAGE SECTION of your program, the calls can be made from the PROCEDURE DIVISION:

```
CALL "CMCL" USING MCL-REGS MCL-service.
```

(Where "MCL-service" is the service call name, shown in SMCLS.IN.C in Appendix C.)

```
CALL "POINT" USING structure structure-ptr.
```

(Where "structure" is a Group-item, and "structure-ptr" is a COMP-2 field, as defined in the WORKING-STORAGE SECTION of the program.)

Example

This example demonstrates how a COBOL program could create a MOD 400 temporary file. Note that the example shows the program after compilation, so the copy files (specified by the COPY statements) now appear in the program.

```
*-------------------------
 WORKING-STORAGE  SECTION.
*-------------------------
* COPY  REGS.

 01   MCL-REGS.
*-------------------------
*
      05   R-B7          USAGE  COMP-2  VALUE  0.
      05   R-B6          USAGE  COMP-2  VALUE  0.
      05   R-B5          USAGE  COMP-2  VALUE  0.
      05   R-B4          USAGE  COMP-2  VALUE  0.
      05   R-B3          USAGE  COMP-2  VALUE  0.
      05   R-B2          USAGE  COMP-2  VALUE  0.
      05   R-B1          USAGE  COMP-2  VALUE  0.
*
      05   R-R7          USAGE  COMP-1  VALUE  0.
      05   R-R6          USAGE  COMP-1  VALUE  0.
      05   R-R5          USAGE  COMP-1  VALUE  0.
```

```
           05  R-R4          USAGE COMP-1 VALUE 0.
           05  R-R3          USAGE COMP-1 VALUE 0.
           05  R-R2          USAGE COMP-1 VALUE 0.
           05  R-R1          USAGE COMP-1 VALUE 0.

       * COPY SCREATE.

        01  CREATE-FILE-TABLE.

            20  CREATE-LFN                    COMP-1 VALUE 1.
            20  CREATE-PATH-PTR               COMP-2.
            20  CREATE-ORGANIZATION           PIC X(02) VALUE "R"1"".
            20  CREATE-RECORD-SIZE            COMP-1 VALUE 58.
            20  CREATE-CI-SIZE                COMP-1 VALUE 1024.
            20  CREATE-INITIAL-ALLOC          COMP-1 VALUE 10.
            20  CREATE-GROWTH-SIZE            COMP-1 VALUE 0.
            20  CREATE-MAXIMUM-SIZE           COMP-1 VALUE 0.
            20  CREATE-FREE-SPACE             COMP-1 VALUE 0.
            20  CREATE-OVERFLOW               COMP-1 VALUE 0.
            20  CREATE-RECORD-DESCRIPTORS     COMP-1 VALUE 0.
            20  CREATE-RECORD-DESCP-PTRS      COMP-2 VALUE 0.
            20  CREATE-RFU-1                  COMP-2 VALUE 0.
            20  CREATE-RFU-2                  COMP-2 VALUE 0.

       01   CREATE-TABLE-PTR                  COMP-2.

       01   TEMP-PATH                         PIC X(02) VALUE SPACE.
       01   TEMP-PATH-PTR                     COMP-2.

       01   MCL-CRFIL                         PIC X(02) VALUE
                                               ""17,49"".

       *-----------------
        PROCEDURE DIVISION.
       *-----------------

        THE-BEGINNING.

          CALL "POINT" USING CREATE-FILE-TABLE CREATE-TABLE-PTR.
          CALL "POINT" USING TEMP-PATH TEMP-PATH-PTR.
          MOVE TEMP-PATH-PTR TO CREATE-PATH-PTR.

        THE-CREATE.

          MOVE CREATE-TABLE-PTR TO R-B4.
          CALL "CMCL" USING MCL-REGS MCL-CRFIL.

       *-------------------------------------------------------
```

Example Description

1. The two calls to POINT return the pointers of the Create File Structure and the temporary path name.
2. The Create File Structure requires that the pointer to the pathname be provided, thus the MOVE will place the TEMP-PATH-PTR in the structure.

3. This Service Call requires the address of the Create File Structure to be in Base Register 4 (R-B4).

4. The call to CMCL will then execute the Service Request "Create File". The return status will be returned in Register R1 (R-R1). Notice that the temporary pathname (TEMP-PATH) is a space which is the requirement for a temporary file. This call will create a RELATIVE file with a logical record size of 58 characters, a block size of 1024, and an initial size of 10. Also, the file will have the GET performed for LFN 1. The user can then use standard COBOL I/O verbs to manipulate the file and when the close is issued, the file will be automatically REMOVEd and deleted.

5. Note that the service call is defined as " "17,49" " which will generate the hexadecimal value of x'1030'. COBOL uses the decimal value as an index to the HEX table where index 1 = 0, etc. See the *GCOS 6 Advanced COBOL Compiler User's Guide* (Order No. CZ31) for further explanation.

Resource Requirements

MEMORY
CMCL shared procedure
24 words

POINT shared procedure
8 words

GETARG

Function

Passes positional arguments to executing COBOL programs so they can function as commands. Any number of arguments may be passed using a single call. (See ARGTAB for passing floating-option arguments.)

Subroutine Syntax

Add to the WORKING-STORAGE SECTION of your Program:

```
01 ARGUMENT-1          PICTURE X(10).
01 ARGUMENT-2          PICTURE X(30).
01 ARGUMENT-3          PICTURE X(5).
01 ARGUMENT-n          .....
```

Add to the PROCEDURE DIVISION of your Program:

```
CALL "GETARG" USING ARGUMENT-1 ARGUMENT-2 ... ARGUMENT-n.
```

Note that GETARG may be called once only during the execution of a single COBOL program. Also, you can name the fields as you like, as long as you use the same names in the CALL statement.

Example

When GETARG is used, each argument must be an 01 level description. GETARG places each argument (taken as a whole entity) in each successive field mentioned in the CALL statement. This is repeated for each argument in the command line.

For an entry of:

```
CBLPRG PRINT>USER>COBOL>CBLPRG.L 123
```

(Assume that CBLPRG calls the GETARG subroutine.)

GETARG, using the WORKING-STORAGE SECTION description shown above (in "Subroutine Syntax"), will pass the arguments into their fields as follows:

ARGUMENT-1 will contain 'PRINT'
ARGUMENT-2 will contain '>USER>COBOL>CBLPRG.L'
ARGUMENT-3 will contain '123'

Notes

GETARG will return only the actual number of arguments passed; you should initialize the arguments before calling GETARG to determine the actual number of arguments passed. Also, GETARG will truncate the actual argument if the receiving field is too small.

GETARG will also function properly if the program is a lead task.

POINT:

Function

Identifies the physical address of a field in memory, as required by certain MOD 400 System Services. POINT is used in conjunction with the CMCL subroutine (see CMCL in this section) and the COBOL copy files included in the appendixes in the back of this manual.

Subroutine Description

See CMCL (in this section) for a description of how POINT, CMCL, and the COBOL copy files work together.

Subroutine Syntax

CALL "POINT" USING structure structure-ptr.

(Where "structure" is a group-item and "structure-ptr" is a COMP-2 field.)

Resource Requirements

MEMORY	Shared procedure	8 words

Appendix A
DPS 6 "R" AND "B" Register Description COBOL Copy File

REGS.IN.C
(COBOL COPY FILE)

All service calls require register descriptions, so include the statement:

```
COPY REGS.
```

in the WORKING-STORAGE SECTION of your program. During program compilation, the following register description will be copied into your program:

```
01    MCL-REGS.
        02    R-B7    USAGE COMP-2 VALUE 0.
        02    R-B6    USAGE COMP-2 VALUE 0.
        02    R-B5    USAGE COMP-2 VALUE 0.
        02    R-B4    USAGE COMP-2 VALUE 0.
        02    R-B3    USAGE COMP-2 VALUE 0.
        02    R-B2    USAGE COMP-2 VALUE 0.
        02    R-B1    USAGE COMP-2 VALUE 0.
*
        02    R-R7    USAGE COMP-1 VALUE 0.
        02    R-R6    USAGE COMP-1 VALUE 0.
        02    R-R5    USAGE COMP-1 VALUE 0.
        02    R-R4    USAGE COMP-1 VALUE 0.
        02    R-R3    USAGE COMP-1 VALUE 0.
        02    R-R2    USAGE COMP-1 VALUE 0.
        02    R-R1    USAGE COMP-1 VALUE 0.
```

Appendix B
COBOL Copy Files for Five Commonly-Used Data Descriptions

SCREATE.IN.C (Create File Information)

If a service call requires a data description for Create File Information, include the statement:

```
COPY SCREATE.
```

in the WORKING-STORAGE SECTION of your program. During program compilation, the following data description will be copied into your program:

```
01   CREATE-FILE-TABLE.
     20   CREATE-LFN                          COMP-1.
     20   CREATE-PATH-PTR                     COMP-2.
     20   CREATE-ORGANIZATION                 PIC X(02)  VALUE "R" "1".
     20   CREATE-RECORD-SIZE                  COMP-1     VALUE 58.
     20   CREATE-CI-SIZE          COMP-1      VALUE 1024.
     20   CREATE-INITIAL-ALLOC                COMP-1     VALUE 10.
     20   CREATE-GROWTH-SIZE                  COMP-1     VALUE 0.
     20   CREATE-MAXIMUM-SIZE                 COMP-1     VALUE 0.
     20   CREATE-FREE-SPACE                   COMP-1     VALUE 0.
     20   CREATE-OVERFLOW                     COMP-1     VALUE 0.
     20   CREATE-RECORD-DESCRIPTORS           COMP-1     VALUE 0.
     20   CREATE-RECORD-DESCP-PTRS            COMP-2     VALUE 0.
     20   CREATE-RFU-1            COMP-2      VALUE 0.
     20   CREATE-RFU-2            COMP-2      VALUE 0.
```

SFIB.IN.C
(File Information Block)

If a service call requires a data description for File Information Block, include the statement:

```
COPY SFIB.
```

in the WORKING-STORAGE SECTION of your program. During program compilation, the following data description will be copied into your program:

```
01   FIB-TABLE.
     05   F-LFN                          COMP-1     VALUE -1.
     05   F-PROV                         PIC X(02)  VALUE " "65,1" ".
     05   F-URP                          COMP-2     VALUE  0.
     05   F-IRL                          COMP-1     VALUE +256.
     05   F-ORL                          COMP-1     VALUE +256.
     05   F-RFU1                         COMP-1     VALUE  0.
     05   F-IRT                          COMP-1     VALUE  0.
     05   F-ORT                          COMP-1     VALUE  0.
     05   F-IKP                          COMP-2     VALUE  0.
     05   F-IKF                          COMP-1     VALUE +0.
     05   F-ORA                          COMP-2     VALUE  0.
     05   F-RFU2                         COMP-2     VALUE  0.

*
*
 01   F-PROV-READ                        PIC X(02)  VALUE " "65,1" ".
 01   F-PROV-REWRITE                     PIC X(02)  VALUE " "121,1" ".
 01   F-PROV-READKEY                     PIC X(02)  VALUE " "69,1" ".
 01   F-PROV-READKEY-RELATIVE            PIC X(02)  VALUE " "65,129" ".
 01   F-PROV-REWRITEKEY                  PIC X(02)  VALUE " "125,1" ".
 01   F-PROV-REWRITEKEY-RELATIVE         PIC X(02)  VALUE " "121,129" ".
```

SINFO.IN.C
(File Information)

If a service call requires a data description for File Information, include the statement:

```
COPY SINFO.
```

in the WORKING-STORAGE SECTION of your program. During program compilation, the following data description will be copied into your program:

```
01   INFO-TABLE.
     20    INFO-LFN-X.
             30    INFO-LFN                 COMP-1      VALUE  0.
     20    INFO-PATH-PTR                   COMP-2      VALUE  0.
     20    INFO-DEVICE-TYPE                COMP-1      VALUE  0.
     20    INFO-LRN                        COMP-1      VALUE  0.
     20    INFO-FILE-TYPE                  PIC X(01)   VALUE SPACE.
     20    INFO-FILE-TYPE-2                PIC X(01)   VALUE SPACE.
     20    INFO-ATTRIBUTE-PTR              COMP-2      VALUE  0.
     20    INFO-DISK-OPTIONS               COMP-1      VALUE  0.
     20    INFO-DISK-ATTRIBUTES            COMP-1      VALUE  0.
     20    INFO-KEY-PTR                    COMP-2      VALUE  0.
     20    INFO-RECORD-DESC-SIZE           COMP-1      VALUE  0.
     20    INFO-NO-RELATED-FILES           COMP-1      VALUE  0.

01   INFO-FILE-ATTR.
     20    INFO-LRSZ                       COMP-1.
     20    INFO-CISZ                       COMP-1.
     20    INFO-USED                       COMP-1.
     20    INFO-GROWTH                     COMP-1.
     20    INFO-MAX                        COMP-1.
     20    INFO-FREE-SPACE                 COMP-1.
     20    INFO-FILLER                     PIC X(20).

01   INFO-REC-KEY.
     20    FILLER                          PIC X(10).
     20    INFO-SECTORS-PER-LOG            COMP-1.
     20    FILLER                          PIC X(06).
```

SGET.IN.C
(Get File Information)

If a service call requires a data description for Get File Information, include the statement:

```
COPY SGET.
```

in the WORKING-STORAGE SECTION of your program. During program compilation, the following data description will be copied into your program:

```
01   GET-FILE-TABLE.
     10   THE-LFN                    COMP-1.
     10   THE-PATH                   COMP-2.
     10   THE-ACCESS-SHARE                    PIC X(02).
     10   TAPE-PARAMETERS.
          20   FILLER                         COMP-1    VALUE  0.
          20   FILLER                         COMP-1    VALUE  0.
          20   FILLER                         COMP-1    VALUE  0.
          20   FILLER                         COMP-1    VALUE  0.
          20   FILLER                         COMP-1    VALUE  0.
          20   FILLER                         COMP-1    VALUE  0.
          20   FILLER                         COMP-1    VALUE  0.

01   ACCESS-RR          PIC X(02)   VALUE " "2,1" ".
01   ACCESS-RW          PIC X(02)   VALUE " "3,1" ".
01   ACCESS-EX          PIC X(02)   VALUE " "4,1" ".
01   ACCESS-WR          PIC X(02)   VALUE " "5,1" ".
01   ACCESS-WW          PIC X(02)   VALUE " "6,1" ".
```

SPATH.IN.C
Expand Path Table

If a service call requires a data description for Expand Path Information, include the statement:

```
COPY SPATH.
```

in the WORKING-STORAGE SECTION of your program. During program compilation, the following data description will be copied into your program:

```
01   EXPAND-PATH-PARAMETERS.
     20   INPUT-PATH-PTR          COMP-2.
     20   OUTPUT-PATH-PTR         COMP-2.
     20   EXPAND-TYPE             COMP-1      VALUE  0.
```

Appendix C
Data Descriptions for System Services

SMCLS.IN.C
(Data Descriptions for System Services)

All MOD 400 System Service Calls require an MCL description in the program's WORKING-STORAGE SECTION to identify the service call name and describe its data field.

MCL descriptions for all MOD 400 System Service Calls are given below. Refer to the *MOD 400 System Programmer's Guide- Volume II* (Order No. CZ06) for complete information on System Service Calls.

```
01  MCL-ABGRP      PIC X(02)  VALUE  " "14,11" ".
01  MCL-ABGRQ      PIC X(02)  VALUE  " "14,08" ".
01  MCL-ACTID      PIC X(02)  VALUE  " "21,03" ".
01  MCL-ACTVG      PIC X(02)  VALUE  " "14,10" ".
01  MCL-ASFIL      PIC X(02)  VALUE  " "17,17" ".
01  MCL-BUAT       PIC X(02)  VALUE  " "13,10" ".
01  MCL-BUID       PIC X(02)  VALUE  " "21,07" ".
01  MCL-BULD       PIC X(02)  VALUE  " "13,11" ".
01  MCL-BUXFR      PIC X(02)  VALUE  " "13,08" ".
01  MCL-CANRQ      PIC X(02)  VALUE  " "13,02" ".
01  MCL-CIN        PIC X(02)  VALUE  " "09,03" ".
01  MCL-CKPFL      PIC X(02)  VALUE  " "14,18" ".
01  MCL-CKPT       PIC X(02)  VALUE  " "14,16" ".
01  MCL-CLFIL      PIC X(02)  VALUE  " "17,86" ".
01  MCL-CLFIL2     PIC X(02)  VALUE  " "17,87" ".
01  MCL-CLFIL3     PIC X(02)  VALUE  " "17,88" ".
01  MCL-CLPNT      PIC X(02)  VALUE  " "13,20" ".
01  MCL-CLRSW      PIC X(02)  VALUE  " "12,03" ".
01  MCL-CMDLN      PIC X(02)  VALUE  " "13,09" ".
01  MCL-CMSUP      PIC X(02)  VALUE  " "10,03" ".
01  MCL-CMSUP2     PIC X(02)  VALUE  " "10,04" ".
01  MCL-CNCRQ      PIC X(02)  VALUE  " "06,02" ".
01  MCL-CNSRQ      PIC X(02)  VALUE  " "07,02" ".
01  MCL-CRDIR      PIC X(02)  VALUE  " "17,161" ".
01  MCL-CRFIL      PIC X(02)  VALUE  " "17,49" ".
01  MCL-CRGRP      PIC X(02)  VALUE  " "14,04" ".
```

SMCLS.IN.C (continued)

```
01  MCL-CROAT      PIC X(02)  VALUE  " "08,11" ".
01  MCL-CRSEG      PIC X(02)  VALUE  " "13,13" ".
01  MCL-CRTSK      PIC X(02)  VALUE  " "13,03" ".
01  MCL-CRTSK2     PIC X(02)  VALUE  " "13,04" ".
01  MCL-CWDIR      PIC X(02)  VALUE  " "17,197" ".
01  MCL-DFCKP      PIC X(02)  VALUE  " "13,26" ".
01  MCL-DFRHD      PIC X(02)  VALUE  " "02,14" ".
01  MCL-DFRTL      PIC X(02)  VALUE  " "02,13" ".
01  MCL-DFSM       PIC X(02)  VALUE  " "07,05" ".
01  MCL-DLDIR      PIC X(02)  VALUE  " "17,86" ".
01  MCL-DLFIL      PIC X(02)  VALUE  " "17,54" ".
01  MCL-DLGRP      PIC X(02)  VALUE  " "14,05" ".
01  MCL-DLOAT      PIC X(02)  VALUE  " "08,14" ".
01  MCL-DLREC      PIC X(02)  VALUE  " "18,49" ".
01  MCL-DLREC2     PIC X(02)  VALUE  " "18,50" ".
01  MCL-DLSEG      PIC X(02)  VALUE  " "13,14" ".
01  MCL-DLSM       PIC X(02)  VALUE  " "07,08" ".
01  MCL-DLTSK      PIC X(02)  VALUE  " "13,05" ".
01  MCL-DQPST      PIC X(02)  VALUE  " "02,12" ".
01  MCL-DSFIL      PIC X(02)  VALUE  " "17,22" ".
01  MCL-DSTRP      PIC X(02)  VALUE  " "11,03" ".
01  MCL-ELEND      PIC X(02)  VALUE  " "03,10" ".
01  MCL-ELEX       PIC X(02)  VALUE  " "03,08" ".
01  MCL-ELGT       PIC X(02)  VALUE  " "03,09" ".
01  MCL-ELST       PIC X(02)  VALUE  " "03,06" ".
01  MCL-ENTID      PIC X(02)  VALUE  " "21,08" ".
01  MCL-ENTRP      PIC X(02)  VALUE  " "11,02" ".
01  MCL-EROUT      PIC X(02)  VALUE  " "09,04" ".
01  MCL-EXTDT      PIC X(02)  VALUE  " "06,05" ".
01  MCL-EXTET      PIC X(02)  VALUE  " "06,14" ".
01  MCL-EXTIM      PIC X(02)  VALUE  " "06,06" ".
01  MCL-GAFIL      PIC X(02)  VALUE  " "11,125" ".
01  MCL-GDTM       PIC X(02)  VALUE  " "06,07" ".
01  MCL-GIDEV      PIC X(02)  VALUE  " "17,103" ".
01  MCL-GIFIL      PIC X(02)  VALUE  " "17,97" ".
01  MCL-GMEM       PIC X(02)  VALUE  " "05,03" ".
01  MCL-GMEM2      PIC X(02)  VALUE  " "05,04" ".
01  MCL-GNFIL      PIC X(02)  VALUE  " "17,61" ".
01  MCL-GRFIL      PIC X(02)  VALUE  " "17,57" ".
01  MCL-GTACT      PIC X(02)  VALUE  " "17,67" ".
01  MCL-GTFIL      PIC X(02)  VALUE  " "17,33" ".
01  MCL-GWDIR      PIC X(02)  VALUE  " "17,193" ".
01  MCL-HDIR       PIC X(02)  VALUE  " "21,12" ".
01  MCL-INDTM      PIC X(02)  VALUE  " "06,08" ".
01  MCL-INSID      PIC X(02)  VALUE  " "21,06" ".
01  MCL-KILLT      PIC X(02)  VALUE  " "13,18" ".
01  MCL-LKNME      PIC X(02)  VALUE  " "17,72" ".
01  MCL-MACPT      PIC X(02)  VALUE  " "22,02" ".
01  MCL-MCME       PIC X(02)  VALUE  " "22,07" ".
01  MCL-MCMG       PIC X(02)  VALUE  " "22,08" ".
01  MCL-MDFIL      PIC X(02)  VALUE  " "17,66" ".
01  MCL-MINIT      PIC X(02)  VALUE  " "22,03" ".
01  MCL-MODID      PIC X(02)  VALUE  " "21,04" ".
01  MCL-MRECV      PIC X(02)  VALUE  " "22,04" ".
01  MCL-MSEND      PIC X(02)  VALUE  " "22,06" ".
```

SMCLS.IN.C (continued)

```
01  MCL-MTMG      PIC X(02)  VALUE  " "22,05" ".
01  MCL-NCIN      PIC X(02)  VALUE  " "09,07" ".
01  MCL-NMLF      PIC X(02)  VALUE  " "09,09" ".
01  MCL-NPROC     PIC X(02)  VALUE  " "14,12" ".
01  MCL-NUIN      PIC X(02)  VALUE  " "09,05" ".
01  MCL-NUOUT     PIC X(02)  VALUE  " "09,06" ".
01  MCL-OPFIL     PIC X(02)  VALUE  " "17,81" ".
01  MCL-OPFIL2    PIC X(02)  VALUE  " "17,82" ".
01  MCL-OPMSG     PIC X(02)  VALUE  " "10,01" ".
01  MCL-OPRSP     PIC X(02)  VALUE  " "10,02" ".
01  MCL-OVEXC     PIC X(02)  VALUE  " "08,01" ".
01  MCL-OVLD      PIC X(02)  VALUE  " "08,02" ".
01  MCL-OVRCL     PIC X(02)  VALUE  " "08,08" ".
01  MCL-OVRLD     PIC X(02)  VALUE  " "08,17" ".
01  MCL-OVRLS     PIC X(02)  VALUE  " "08,07" ".
01  MCL-OVRSV     PIC X(02)  VALUE  " "08,06" ".
01  MCL-OVST      PIC X(02)  VALUE  " "08,04" ".
01  MCL-OVUN      PIC X(02)  VALUE  " "08,13" ".
01  MCL-PERID     PIC X(02)  VALUE  " "21,02" ".
01  MCL-PPNTL     PIC X(02)  VALUE  " "02,15" ".
01  MCL-PRFAU     PIC X(02)  VALUE  " "37,67" ".
01  MCL-PRFCR     PIC X(02)  VALUE  " "37,33" ".
01  MCL-PRFDL     PIC X(02)  VALUE  " "37,49" ".
01  MCL-PRFGT     PIC X(02)  VALUE  " "37,17" ".
01  MCL-PRFIF     PIC X(02)  VALUE  " "37,19" ".
01  MCL-PRFUP     PIC X(02)  VALUE  " "37,65" ".
01  MCL-RBADD     PIC X(02)  VALUE  " "02,08" ".
01  MCL-RBOOT     PIC X(02)  VALUE  " "33,07" ".
01  MCL-RBPRM     PIC X(02)  VALUE  " "33,06" ".
01  MCL-RCLHD     PIC X(02)  VALUE  " "02,16" ".
01  MCL-RDBLK     PIC X(02)  VALUE  " "18,01" ".
01  MCL-RDBLK2    PIC X(02)  VALUE  " "18,02" ".
01  MCL-RDBLK3    PIC X(02)  VALUE  " "18,03" ".
01  MCL-RDBLK4    PIC X(02)  VALUE  " "18,04" ".
01  MCL-RDBLK5    PIC X(02)  VALUE  " "18,05" ".
01  MCL-RDREC     PIC X(02)  VALUE  " "18,17" ".
01  MCL-RDREC2    PIC X(02)  VALUE  " "18,18" ".
01  MCL-RDREC3    PIC X(02)  VALUE  " "18,19" ".
01  MCL-RDREC4    PIC X(02)  VALUE  " "18,20" ".
01  MCL-RDREC5    PIC X(02)  VALUE  " "18,21" ".
01  MCL-RDREC6    PIC X(02)  VALUE  " "18,22" ".
01  MCL-RDREC7    PIC X(02)  VALUE  " "18,23" ".
01  MCL-RDREC8    PIC X(02)  VALUE  " "18,26" ".
01  MCL-RDSW      PIC X(02)  VALUE  " "12,01" ".
01  MCL-RLDMP     PIC X(02)  VALUE  " "33,05" ".
01  MCL-RLSM      PIC X(02)  VALUE  " "07,04" ".
01  MCL-RLTML     PIC X(02)  VALUE  " "24,05" ".
01  MCL-RMEM      PIC X(02)  VALUE  " "05,05" ".
01  MCL-RMEM2     PIC X(02)  VALUE  " "05,06" ".
01  MCL-RMFIL     PIC X(02)  VALUE  " "17,38" ".
01  MCL-RNFIL     PIC X(02)  VALUE  " "17,65" ".
01  MCL-ROLBK     PIC X(02)  VALUE  " "13,21" ".
01  MCL-RPDFC     PIC X(02)  VALUE  " "16,05" ".
01  MCL-RPMSG     PIC X(02)  VALUE  " "16,04" ".
01  MCL-RQBAT     PIC X(02)  VALUE  " "15,01" ".
```

SMCLS.IN.C (continued)

```
01  MCL-RQCL      PIC X(02)  VALUE  " "06,01" ".
01  MCL-RQGRP     PIC X(02)  VALUE  " "14,01" ".
01  MCL-RQIO      PIC X(02)  VALUE  " "03,01" ".
01  MCL-RQSM      PIC X(02)  VALUE  " "07,01" ".
01  MCL-RQSPT     PIC X(02)  VALUE  " "24,03" ".
01  MCL-RQTML     PIC X(02)  VALUE  " "24,04" ".
01  MCL-RQTSK     PIC X(02)  VALUE  " "13,01" ".
01  MCL-RS        PIC X(02)  VALUE  " "14,17" ".
01  MCL-RSTID     PIC X(02)  VALUE  " "17,06" ".
01  MCL-RSVSM     PIC X(02)  VALUE  " "07,03" ".
01  MCL-RWREC     PIC X(02)  VALUE  " "18,65" ".
01  MCL-RWREC2    PIC X(02)  VALUE  " "18,66" ".
01  MCL-RVFPW     PIC X(02)  VALUE  " "37,02" ".
01  MCL-SDL       PIC X(02)  VALUE  " "28,01" ".
01  MCL-SETSW     PIC X(02)  VALUE  " "12,02" ".
01  MCL-SGRPA     PIC X(02)  VALUE  " "14,20" ".
01  MCL-SGTRP     PIC X(02)  VALUE  " "11,04" ".
01  MCL-SHCS      PIC X(02)  VALUE  " "13,38" ".
01  MCL-SHFIL     PIC X(02)  VALUE  " "17,56" ".
01  MCL-SHGWS     PIC X(02)  VALUE  " "14,23" ".
01  MCL-SPGRP     PIC X(02)  VALUE  " "14,06" ".
01  MCL-SPTSK     PIC X(02)  VALUE  " "13,06" ".
01  MCL-SPTSK2    PIC X(02)  VALUE  " "13,07" ".
01  MCL-SPTSK3    PIC X(02)  VALUE  " "13,22" ".
01  MCL-STMP      PIC X(02)  VALUE  " "05,07" ".
01  MCL-STTID     PIC X(02)  VALUE  " "17,01" ".
01  MCL-STTY      PIC X(02)  VALUE  " "17,70" ".
01  MCL-SUSPG     PIC X(02)  VALUE  " "14,09" ".
01  MCL-SUSPN     PIC X(02)  VALUE  " "06,03" ".
01  MCL-SUSPN2    PIC X(02)  VALUE  " "06,04" ".
01  MCL-SWFIL     PIC X(02)  VALUE  " "17,91" ".
01  MCL-SYSAT     PIC X(02)  VALUE  " "21,18" ".
01  MCL-SYSID     PIC X(02)  VALUE  " "21,05" ".
01  MCL-TEST      PIC X(02)  VALUE  " "02,03" ".
01  MCL-TGIN      PIC X(02)  VALUE  " "21,13" ".
01  MCL-TIFIL     PIC X(02)  VALUE  " "17,99" ".
01  MCL-TOFIL     PIC X(02)  VALUE  " "17,100" ".
01  MCL-TRMRQ     PIC X(02)  VALUE  " "02,04" ".
01  MCL-TRMRQ2    PIC X(02)  VALUE  " "02,05" ".
01  MCL-TRPHD     PIC X(02)  VALUE  " "11,01" ".
01  MCL-ULNME     PIC X(02)  VALUE  " "17,73" ".
01  MCL-USIN      PIC X(02)  VALUE  " "09,01" ".
01  MCL-USOUT     PIC X(02)  VALUE  " "09,02" ".
01  MCL-USRID     PIC X(02)  VALUE  " "21,01" ".
01  MCL-VLCKP     PIC X(02)  VALUE  " "14,19" ".
01  MCL-XPCS      PIC X(02)  VALUE  " "13,37" ".
01  MCL-WAIT      PIC X(02)  VALUE  " "02,01" ".
01  MCL-WAITA     PIC X(02)  VALUE  " "02,02" ".
01  MCL-WAITL     PIC X(02)  VALUE  " "02,02" ".
01  MCL-WAITM     PIC X(02)  VALUE  " "02,02" ".
01  MCL-WIFIL     PIC X(02)  VALUE  " "17,101" ".
01  MCL-WOFIL     PIC X(02)  VALUE  " "17,102" ".
01  MCL-WRBLK     PIC X(02)  VALUE  " "18,17" ".
01  MCL-WRBLK2    PIC X(02)  VALUE  " "18,1" ".
01  MCL-WRREC     PIC X(02)  VALUE  " "17,33" ".
```

SMCLS.IN.C (continued)

```
01   MCL-WRREC2          PIC X(02)    VALUE     " "17,34" ".
01   MCL-WRREC3          PIC X(02)    VALUE     " "17,35" ".
01   MCL-WRREC4          PIC X(02)    VALUE     " "17,36" ".
01   MCL-WRREC5          PIC X(02)    VALUE     " "17,37" ".
01   MCL-WRREC6          PIC X(02)    VALUE     " "17,38" ".
01   MCL-WRREC7          PIC X(02)    VALUE     " "17,39" ".
01   MCL-WTBLK           PIC X(02)    VALUE     " "18,33" ".
01   MCL-XFERU           PIC X(02)    VALUE     " "24,07" ".
01   MCL-XPATH           PIC X(02)    VALUE     " "17,209" ".
01   MCL-XRETU           PIC X(02)    VALUE     " "24,08" ".
```

HONEYWELL INFORMATION SYSTEMS
Technical Publications Remarks Form

TITLE: DPS 6
TOOLKIT SERIES-I
REFERENCE MANUAL

ORDER NO. HH06-00

DATED JUNE 1986

ERRORS IN PUBLICATION

SUGGESTIONS FOR IMPROVEMENT TO PUBLICATION

Your comments will be investigated by appropriate technical personnel and action will be taken as required. Receipt of all forms will be acknowledged; however, if you require a detailed reply, check here. ☐

FROM: NAME _____ DATE _____

TITLE _____

COMPANY _____

ADDRESS _____

_____

CUT ALONG LINE

PLEASE FOLD AND TAPE–
NOTE: U.S. Postal Service will not deliver stapled forms

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 39531 WALTHAM, MA 02154

POSTAGE WILL BE PAID BY ADDRESSEE

HONEYWELL INFORMATION SYSTEMS
200 SMITH STREET
WALTHAM, MA 02154

ATTN: PUBLICATIONS, MS486

Honeywell